Newton Method in 3-Dimensional Shape Optimization Problems

Arian Novruzi, Jean R. Roche

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Newton method in 3-dimensional Shape Optimization Problems.

Arian Novruzi and Jean R. Roche

N° 3367

Février 1998

THÈME 4

ISSN 0249-6399

INRIA
LORRAINE

Newton method in 3-dimensional Shape Optimization Problems.

Arian Novruzi * and Jean R. Roche †

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet Numath

Rapport de recherche n ° 3367 — Février 1998 — 33 pages

Abstract: Our goal is to introduce a Newton method to compute the stationary points of a total energy with respect to the shape. We produce a precise description of the second order shape derivative which is given by a symmetrical boundary integral operator, useful for numerical calculations. We apply this method to a particular shape optimization problem, the electromagnetic casting problem. The algorithm to compute the shape gradient and the shape Hessian is adapted to a MIMD computer with distributed memory using M.P.I. message passing interface.

Key-words: shape optimization, Newton method, electromagnetic casting.

(Résumé : tsvp)

* Université Henri Poincaré, CNRS UMR 9973 & INRIA Lorraine, Projet Numath, B.P. 239, 54506 Vandoeuvre les Nancy Cedex, France
† Université Henri Poincaré, CNRS UMR 9973 & INRIA Lorraine, Projet Numath, B.P. 239, 54506 Vandoeuvre les Nancy Cedex, France, e-mail: roche@loria.fr

Unité de recherche INRIA Lorraine
Technopôle de Nancy-Brabois, Campus scientifique,
615 rue de Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY (France)
Téléphone : (33) 83 59 30 30 – Télécopie : (33) 83 27 83 19
Antenne de Metz, technopôle de Metz 2000, 4 rue Marconi, 55070 METZ
Téléphone : (33) 87 20 35 00 – Télécopie : (33) 87 76 39 77

La méthode de Newton en optimisation de formes tridimensionelle.

Résumé : L'objet de ce travail est d'introduire la méthode de Newton dans le calcul d'un point critique d'une énergie par rapport à la forme. Nous donnons ici une description précise de la dérivée seconde qui est un opérateur intégral sur le bord, adapté au calcul numérique. On applique cette méthode à un problème de formage électromagnétique en trois dimensions. L'algorithme de calcul du gradient et de l'Hessien par rapport à la forme est adapté à un ordinateur de type MIMD avec mémoire distribuée.

Mots-clé : optimisation de formes, méthode de Newton, magnétoformage.

1 Introduction.

In numerical simulation of electromagnetic casting one approach is to consider models where the computation of the free boundary amounts to solve a shape optimization problem. The functional to be minimized is the total energy of the phenomenon under consideration.

Typically we want to compute a shape Ω^* such that

$$\Omega^* = \text{argmin}\{E(\Omega) : \Omega \in \mathcal{O}\} \tag{1}$$

where $\mathcal{O}$ is a set of admissible domains and E is a cost function depending on the solution of a partial differential elliptic system. Our interest is to obtain numerically by a Newton method a point Ω^* which satisfies the Kuhn-Tucker conditions. To compute first and second order shape derivatives we introduce an appropriate Banach space, see [9], [12], [13] and [4].

The simplified model of the 3-dimensional electromagnetic shaping problem studied here concerns a bubble of liquid metal levitating in the electromagnetic field created by given conductors. Under suitable assumptions, the equilibrium configurations of the liquid metal are described by a set of equations involving a relation at the boundary between the electromagnetic, superficial and gravity forces. The equilibrium shape is shown to be the stationary state of the total energy under the constraint that the volume is prescribed. Here the total energy depends also of the solution of an exterior Neumann Problem, see [14] and [21].

In a previous work [15], we introduce classical Quasi-Newton methods associated with penalization techniques to solve our optimization problem. Such methods involve only first derivatives of the cost function and the resolution of an exterior Neumann partial differential system. If Newton's method is used, the computation of the second order shape derivatives is also dominated by the P.D.E. system resolution, the evaluation of the second order derivatives does not change the order of computational complexity.

In [9] we introduce a precise method to compute first and second order shape derivatives. Under suitable regularity assumptions, the first and second order derivatives are boundary integral operators. We then introduce an integral representation of the solution of the exterior Neumann boundary value problem, see [15], [8].

Numericaly, we construct a sequence of domains determinated by their boundaries. We consider piecewise linear surfaces with n nodes. The integral equation is solved by a Galerkin finite element method.

Numerical estimation of the integral equation and the computation of shape Hessian needs $O(n^3)$ floating point operations. Then to minimize the computational time taken by each iteration we introduce a parallel implementation of this sections of our algorithm. To this end we adapt the software to a MIMD computer with distributed memory using M.P.I. messages-passing interface. We present several numerical exemples, where we analyse the iteration evolution and the efficiency of our parallel algorithm.

2 The Electromagnetic casting problem.

We denote by ω an open and bounded domain in $\mathbb{R}^3$ filled by the liquid metal. Let Ω be its exterior and $\Gamma = \partial\omega$ its boundary. We assume that the frequency of the imposed current is sufficiently high so that the magnetic field does not penetrate into the metal and the electromagnetic forces are reduced to the magnetic pressure acting on the interface.
Under the above assumptions, the surface Γ is characterized by the following equilibrium equations.

$$\nabla \wedge B = \mu_0 j_0 \quad \text{in} \quad \Omega \tag{2}$$

$$\nabla \cdot B = 0 \quad \text{in} \quad \Omega \tag{3}$$

$$B \cdot \nu = 0 \quad \text{on} \quad \Gamma \tag{4}$$

$$|B(x)| = O(|x|^{-2}) \quad \text{at} \quad \infty \tag{5}$$

$$\frac{\|B\|^2}{2\mu_0} + \sigma\mathcal{H} + \rho g \cdot x_3 = \Lambda \quad \text{on} \quad \Gamma \tag{6}$$

where j_0 is the current density, B the magnetic field, μ_0 the magnetic permeability, ρ the density of metal, x_3 the height variable, $\mathcal{H}$ the mean curvature of Γ, σ the surface tension and ν the unit normal vector oriented towards Ω. The constant Λ and the surface Γ are the unknowns of the problem.
The total energy of the system (see [15]) is given by :

$$E(\omega) = \frac{-1}{2\mu_0}\int_\Omega B^2 dx + \sigma\int_\Gamma d\gamma(x) + \int_\omega \rho g\, x_3\, dx \tag{7}$$

where B is a solution to (2)-(5).
With some usual hypotheses, a critical point of $E(\omega)$ under the constraint that the volume of ω is prescribed satisfies the nonlinear equilibrium relation (6) which characterizes the boundary Γ.

To compute the magnetic field B we set $B = B^1 + \nabla\varphi$ where B^1 is given by the Biot-Savart law. Then the scalar potential φ is the unique solution of the following exterior Neumann problem:

$$\left\{ \begin{array}{c} -\Delta\varphi = 0 \text{ in } \Omega \\ \dfrac{\partial\varphi}{\partial\nu} = -B^1.\nu \text{ on } \Gamma \\ |\varphi(x)| = o(1) \text{ as } |x| \to \infty \end{array} \right. \tag{8}$$

The solution of (8) can be represented by a single layer integral representation.

THEOREM 2.1 *The solution of the exterior Neumann system (8) is given by the following integral representation:*

$$\varphi(x) = \frac{1}{4\pi} \int_\Gamma \frac{q(y)}{|x-y|} d\gamma(y) \tag{9}$$

where q is the unique solution in $H^{1/2}(\Gamma)$ of the following second kind Fredholm equation, see [10] and [2].

$$-B^1.\nu(x) = \frac{1}{2}q(x) + \frac{1}{4\pi} \int_\Gamma q(y) \frac{\nu(x)\cdot(x-y)}{|x-y|^3} d\gamma(y) \ \forall x \in \Gamma \tag{10}$$

For a proof of the theorem see [15].

3 The Algorithm.

3.1 Shape Derivatives.

Let $Q \in I\!R^3$ be a open and bounded set of class C^∞, locally in one side of ∂Q, see [12].

Let Θ be the set:

$$\Theta = C^k(\overline{Q}; I\!R^3) \tag{11}$$

the set of k-times differentially functions. This set is a Banach space with the standard norm.

In Θ we define the following function ω et Γ:

$$\begin{cases} \omega : \Theta \to \mathcal{P}(I\!R^3) \\ \omega : \theta \to \omega(\theta) = \{\theta(x) : x \in Q\} \end{cases} \tag{12}$$

$$\begin{cases} \Gamma : \Theta \to \mathcal{P}(I\!R^3) \\ \Gamma : \theta \to \Gamma(\theta) = \{\theta(x) : x \in \partial Q\} \end{cases} \tag{13}$$

where $\mathcal{P}(I\!R^3)$ is the set of all subsets of $I\!R^3$. Let $\Omega(\theta)$ be the complementary set in $I\!R^3$ of $\overline{\omega(\theta)}$.

Then we assume that the cost function E is real valued defined on the set:

$$\mathcal{O} = \Big\{\omega \subset I\!R^3 \colon \exists\ \theta \in \mathcal{O}^{-1}, \theta(Q) = \omega\Big\} \tag{14}$$

where $\mathcal{O}^{-1}$ is the subset of Θ given by:

$$\mathcal{O}^{-1} = \Big\{\theta \in \Theta, \text{ is a diffeomorphism of class } C^k \text{ from } \overline{Q} \text{ onto his image}\Big\} \tag{15}$$

Instead of the functional E we consider the cost function G defined in the following way:

$$\begin{cases} G : \mathcal{O}^{-1} \in \Theta \to I\!R \\ G(\theta) = E \circ w(\theta) = E\Big(\omega(\theta)\Big). \end{cases} \tag{16}$$

Now we consider Fréchet derivatives G' and G'' of G, see [11], [12], [4], [5].

DEFINITION 3.1 *If for $\theta \in \mathcal{O}^{-1}$ exits a continuous linear operator of $\mathcal{L}(\mathcal{O}^{-1}, I\!R)$ denoted $\xi \to G'(\theta)(\xi)$ such that:*

$$\begin{cases} G(\theta+\xi) = G(\theta) + G'(\theta)(\xi) + \epsilon(\theta,\xi)\|\xi\|_{\mathcal{O}^{-1}} \\ \lim_{\xi\to 0} \epsilon(\theta,\xi) = 0 \end{cases} \tag{17}$$

$G'(\theta)(\xi)$ is called the first Fréchet derivative of G at θ.

DEFINITION 3.2 *We say that G has a second derivative in $\theta \in \mathcal{O}^{-1}$ if there exists $G"(\theta) \in \mathcal{L}^2(\mathcal{O}^{-1}, I\!R)$, such that:*

$$\begin{cases} G'(\theta+\eta)(\xi) = G'(\theta)(\xi) + G''(\theta)(\xi,\eta) + \epsilon(\theta,\xi,\eta)\|\eta\|_{\mathcal{O}^{-1}} \\ \lim_{\eta\to 0} \epsilon(\theta,\xi,\eta) = 0 \end{cases} \tag{18}$$

In the following of this section we introduce the definition of the shape derivative of functions, in particular of the solution of the exterior Neumann problem(8).

Let $W^2(\Omega(\theta))$ be the closure of $\mathcal{D}(\Omega(\theta))$ for the semi-norm:

$$\phi \to \sum_{i,j} ||\frac{\partial^2 \phi}{\partial x_i x_j}||_{L^2} \tag{19}$$

The exterior Neumann problem (2.7) have a unique solution in $W^2(\Omega(\theta))$, see [19] .

Let be $I\!H$ the set of all the function $u \in W^2(\Omega(\theta))$ for all $\theta \in \mathcal{O}^{-1}$.

$$I\!H = \bigcup_{\theta\in\mathcal{O}^{-1}} W^2(\Omega(\theta)) \tag{20}$$

Let φ the function defined in the following way

$$\begin{array}{l} \varphi : \mathcal{O}^{-1} \to I\!H \\ \varphi : \theta \to \varphi(\theta) \in W^2(\Omega(\theta)) \end{array} \tag{21}$$

the solution of (8) in $\Omega = \Omega(\theta)$.

DEFINITION 3.3 *The function $\varphi : \theta \to \varphi(\theta)$ is called locally differentiable from $\mathcal{O}^{-1}$ in $W^2(\Omega(\theta))$ if for any $\mathcal{U} \subset \Omega(\theta))$ the function $\varphi(\theta)$ restricted to $\mathcal{U}$ is differentiable from $\mathcal{O}^{-1}$ in $W^2(\mathcal{U})$.The value of this derivative in the direction ξ, is noted $\varphi'(\theta)(\xi)$.*

In the next section we introduce the Newton method applied to the Karush, Kuhn and Tucker(KKT) first-order necessary conditions.

3.2 Newton Lagrange method.

To describe the Newton method studied in this paper, we introduce the Lagrangian (see [3]) :

$$\begin{aligned} L(\theta,\Lambda) &= G(\theta)+\Lambda\Big(m(\theta)-m_0\Big) \\ &= -\frac{1}{2\mu_0}L_1(\theta)+\sigma L_2(\theta)+\rho g L_3(\theta)+\Lambda L_4(\theta) \end{aligned} \tag{22}$$

with:

$$L_1(\theta)=\int_{\Omega(\theta)} B^2(\theta)dx \quad , \quad L_2(\theta)=\int_{\Gamma(\theta)} dx \tag{23}$$

$$L_3(\theta)=\int_{\omega(\theta)} x_3 dx \quad , \quad L_4(\theta)=\int_{\omega(\theta)} dx - m_0 \tag{24}$$

where $\theta \in \mathcal{O}^{-1}$, $\Lambda \in I\!R$, $m(\theta)$ is the volume of $\omega(\theta)$ and m_0 is the given volume. A critical point of the energy G with the constraint $m(\theta)-m_0=0$ is the first argument of the couple (θ^*,Λ^*) solution of the following first order KKT necessary conditions:

$$D(\theta,\Lambda)=\begin{pmatrix} L'(\theta,\Lambda) \\ m(\theta)-m_0 \end{pmatrix}=0. \tag{25}$$

The Newton method to compute an approximation of (θ^*,Λ^*) consists in computing a sequence of solutions (θ^k,Λ^k) of a linearized form of (25). This leads to the following algorithm :

$$\begin{cases} \theta^\circ,\Lambda^\circ \text{ given} \\ (\theta^{k+1},\Lambda^{k+1})=(\theta^k,\Lambda^k)+(\delta\theta^k,\delta\Lambda^k) \\ \text{ where } (\delta\theta^k,\delta\Lambda^k)\in \mathcal{O}^{-1}\times I\!R \text{ satisfies} \\ D(\theta^k,\Lambda^k)+H(\theta^k,\Lambda^k)(\delta\theta^k,\delta\Lambda^k)=0 \text{ in } \mathcal{L}(\mathcal{O}^{-1},I\!R)\times I\!R \end{cases} \tag{26}$$

where $H(\theta^k,\Lambda^k)$ is given by:

$$H(\theta^k,\Lambda^k)=\begin{pmatrix} L''(\theta^k,\Lambda^k) & m'(\theta^k) \\ m'(\theta^k) & 0 \end{pmatrix} \tag{27}$$

The computations are carried out for the corresponding discretized formulation of problem (26). The domains $\omega(\theta)$ under consideration are characterized by the boundary $\partial\omega(\theta), = \Gamma$. In practice we consider domains with piecewise linear boundaries.

3.3 Shape derivatives of L.

Given ξ, η, θ in $\mathcal{O}^{-1}$ we introduce vector fields :

$$V(x) = \xi \circ \theta^{-1}(x) \tag{28}$$

and

$$W(x) = \eta \circ \theta^{-1}(x). \tag{29}$$

The i-th component of V is denoted V_i. With this notation the j-th partial derivative of V_i is denoted $D_j V_i$ and DV is the matrix $[D_j V_i]$.

It is also convenient to employ the summation convention, that is that in any expression repeated indices indicate summation from 1 to 3.

For any regular function $g : I\!R^3 \to I\!R$ the vector $\delta g = (\delta_1 g, \delta_2 g, \delta_3 g)$ is defined by:

$$\delta g = Dg - \nu(Dg.\nu) = ([D_i g - \nu_i \nu_j D_j]_{i=1,2,3}) \tag{30}$$

so that δg is merely the component of the gradient of g in the tangential direction at Γ, (see [20]) .

If φ is the solution of the exterior Neumann problem (8), the shape derivative $\varphi'(\theta)(\xi)$ is the solution of (see [12]):

$$\left\{ \begin{array}{c} -\Delta\varphi'(\theta)(\xi) = 0 \text{ in } \Omega \\ \dfrac{\partial\varphi'(\theta)(\xi)}{\partial\nu} = B^1.\delta(V.\nu) - (V.\nu)\left(\dfrac{\partial^2\varphi}{\partial\nu^2} + \dfrac{\partial B^1}{\partial\nu}.\nu\right) \text{ in } \Gamma \\ |\varphi'(\theta)(\xi)(x)| = O(|x|^{-1}) \text{ as } |x| \to \infty \end{array} \right. \tag{31}$$

For simplicity, let us denote the shape derivative $\varphi'(\theta)(\xi)$ by φ'_V.

We compute a solution of (31) by the same technique used to solve the exterior Neumann problem (8).

Using the formal framework introduced in the previous section we compute first and second order derivatives of $L(\theta, \Lambda)$.

THEOREM 3.1 *Let $\theta, \xi, \eta \in \mathcal{O}^{-1}$ and L given in (22). Assume that the derivative $\varphi'(\theta)$ exists. Assume that the current density $j_\circ$ is bounded in $\mathbb{R}^3$ with support in $\Omega(\theta)$. Then :*

$$L'(\theta,\Lambda)(\xi) = \frac{1}{2\mu_0}\int_\Gamma (B^2 + \sigma\mathcal{H} + \rho g x_3 - \Lambda)(V\cdot\nu)d\gamma \tag{32}$$

and

$$\begin{aligned} L''(\theta,\Lambda)(\xi,\eta) &= \int_{\Gamma(\theta)} (\frac{\partial B^2}{\partial \nu} - \mathcal{H}B^2)(V\cdot\nu))(W\cdot\nu)d\gamma \\ &- \int_{\Gamma(\theta)} (\nu\cdot\delta V\cdot W + \nu\cdot\delta W\cdot V + V\cdot\delta\nu\cdot W)B^2 d\gamma \\ &- 2\int_{\Omega(\theta)} D\varphi'(\theta)(\xi)\cdot D\varphi'(\theta)(\eta)dx \\ &+ \sigma\int_\Gamma [(\delta_i V_i)(\delta_i W_i) - \delta_i V_j \delta_j W_i d\gamma \\ &\qquad (\nu\cdot\delta_j V)(\nu\cdot\delta_j W)]\, d\gamma \\ &+ \rho g\int_\Gamma x_3 V_i(\nu_i\delta_j W_j - \nu_j\delta_i W_j) + W_3(V\cdot\nu)d\gamma \\ &+ \Lambda\int_\Gamma V_i(\nu_i\delta_j W_j - \nu_j\delta_i W_j)d\gamma \end{aligned} \tag{33}$$

where V and W are given in (28) and (29).

A complete proof of (32) is given in [6], [7], we give here a sketch of the procedure used to demonstrate (33), more detail can be find in [8].
Proof: The shape derivative of integral functions have been studied by J. Simon [12]. In our context we obtain:

$$\begin{aligned} L_1'(\theta)(\xi) &= -\int_{\Omega(\theta)} 2(D\varphi(\theta) + B^1)\cdot D\varphi'(\theta)(\xi)d\gamma \\ &+ \int_{\Gamma(\theta)} B^2(V\cdot\nu)d\gamma \end{aligned} \tag{34}$$

where $\varphi(\theta)$ satisfy (8) and $\varphi'(\theta)(\xi)$ is solution of (31). Because the Green theorem the first term in the second member is nil, hence,

$$L_1'(\theta)(\xi) = \int_{\Gamma(\theta)} B^2 (V \cdot \nu) d\gamma \tag{35}$$

Using the same technique we obtain the second order shape derivative:

$$\begin{aligned} L_1''(\theta)(\xi,\eta) &= \int_{\Gamma(\theta)} 2B \cdot B'(\theta)(\eta)(V \cdot \nu) + B^2 (V'(\theta)(\eta) \cdot \nu) d\gamma \\ &- \int_{\Gamma(\theta)} B^2 (V \cdot \delta(W \cdot \nu)) d\gamma \\ &+ \int_{\Gamma(\theta)} (W \cdot \nu)(-\mathcal{H} B^2 (V \cdot \nu)) d\gamma \\ &+ \int_{\Gamma(\theta)} (W \cdot \nu)(\frac{\partial B^2}{\partial \nu}(V \cdot \nu) + B^2 (\frac{\partial V}{\partial \nu} \cdot \nu)) d\gamma \end{aligned} \tag{36}$$

Using the notation introduced in the beginning of the section we obtain , see [20]:

$$\begin{aligned} V'(\theta)(\eta) \cdot \nu &= -(DV \cdot W) \cdot \nu = -\nu_i DV_i \cdot W = -\nu_i (\delta V_i + \frac{\partial V_i}{\partial \nu} \nu) \cdot W \\ &= -(\delta V \cdot W) \cdot \nu - (\frac{\partial V}{\partial \nu} \cdot \nu)(W \cdot \nu) \end{aligned} \tag{37}$$

and

$$\begin{aligned} V \cdot \delta(W \cdot \nu) &= V \cdot (\nu_i \delta W_i + W_i \delta \nu_i) = \nu_i (\delta W_i \cdot V) + (V \cdot \delta \nu_i) W_i \\ &= (\delta W \cdot V) \cdot \nu + V \cdot \delta \nu \cdot W \end{aligned} \tag{38}$$

Therefore,

$$\begin{aligned} L_1''(\theta)(\xi,\eta) &= \int_{\Gamma(\theta)} (-\mathcal{H} B^2 + \frac{\partial B^2}{\partial \nu})(V \cdot \nu)(W \cdot \nu) d\gamma \\ &- \int_{\Gamma(\theta)} (\nu \cdot \delta V \cdot W + \nu \cdot \delta W \cdot V + V \cdot \delta \nu \cdot W) B^2 d\gamma \\ &+ \int_{\Gamma(\theta)} 2B \cdot B'(\theta)(\eta)(V \cdot \nu) d\gamma \end{aligned} \tag{39}$$

Note that in $\Gamma(\theta)$:

$$\delta_i D_i\varphi = D_{ii}\varphi - \nu_i\nu_j D_{ji}\varphi = \Delta\varphi - \frac{\partial^2\varphi}{\partial\nu^2} = -\frac{\partial^2\varphi}{\partial\nu^2} \tag{40}$$

$$\delta_i B_i^1 = D_i B_i^1 - \nu_i\nu_j D_j B_i^1 = d\vec{i}v B^1 - \frac{\partial B^1}{\partial\nu}\nu = -\frac{\partial B^1}{\partial\nu}\nu \tag{41}$$

Since:

$$\int_{\Gamma(\theta)} (B\cdot B'(\theta)(\eta))(V\cdot\nu)d\gamma \tag{42}$$
$$= -\int_{\Gamma(\theta)} \varphi'_W((D\varphi + B^1)\cdot\delta(V\cdot\nu) - (V\cdot\nu)(\frac{\partial^2\varphi}{\partial\nu^2} + \frac{\partial B^1}{\partial\nu}\nu))d\gamma$$

we have:

$$\int_{\Gamma(\theta)} (B\cdot B')(V\cdot\nu)d\gamma = -\int_{\Omega(\theta)} D\varphi'_V\cdot D\varphi'_W dx \tag{43}$$

Therefore,

$$\begin{aligned} L_1''(\theta)(\xi,\eta) &= \int_{\Gamma(\theta)} (\frac{\partial B^2}{\partial\nu} - \mathcal{H}B^2)(V\cdot\nu))(W\cdot\nu)d\gamma \\ &- \int_{\Gamma(\theta)} (\nu\cdot\delta V\cdot W + \nu\cdot\delta W\cdot V + V\cdot\delta\nu\cdot W)B^2 d\gamma \\ &- 2\int_{\Omega(\theta)} D\varphi'(\theta)(\xi)\cdot D\varphi'(\theta)(\eta)dx \end{aligned} \tag{44}$$

In [20] E. Giusti proof that for $\theta,\xi,\eta\in\mathcal{O}^{-1}$ we have:

$$L_2'(\theta)(\xi) = \int_{\Gamma(\theta)} \delta_i V_i d\gamma \tag{45}$$

and

$$L_2''(\theta)(\xi,\xi) = \int_{\Gamma(\theta)} (\delta_i V_i)^2 - \delta_i V_j\delta_j V_i + (\nu\cdot\delta_j V)(\nu\cdot\delta_j V)d\gamma \tag{46}$$

Hence using the same technique we obtain:

$$L_2''(\theta)(\xi,\eta) = \int_{\Gamma(\theta)} \delta_i V_i \delta_i W_i - \delta_i V_j \delta_j W_i + (\nu \cdot \delta_j V)(\nu \cdot \delta_j W) d\gamma \tag{47}$$

Introducing a change of variables and if $\theta, \eta \in \mathcal{O}^{-1}$ we have:

$$L_3(\theta + \xi) = \int_{\omega(\theta)} (x_3 + V_3(x)) det[I + DV] dx \tag{48}$$

where $V(x) = (V_1(x), V_2(x), V_3(x)) = \xi \circ \theta^{-1}(x)$. But the shape derivative of $det[I + DV]$ is given by $tr[DV]$ and the second order shape derivative is equal to $tr[DV]tr[DW] - tr[DV.DW]$. Then

$$\begin{aligned} L_3'(\theta)(\xi) &= \int_{\Gamma(\theta)} (V_3(x) + x_3 tr[DV]) dx \\ &= \int_{\Gamma(\theta)} x_3 (V.\nu) d\gamma \end{aligned} \tag{49}$$

and

$$L_3''(\theta)(\xi,\eta) = \int_{\Gamma(\theta)} x_3 V_i (\delta_i W_i \nu_i - \delta_i W_j \nu_j) + W_3 (V.\nu) d\gamma \tag{50}$$

Using the same techniques to compute the shape derivatives of $L_4(\theta)$ we obtain

$$L_4'(\theta)(\xi) = \int_{\omega} tr[DV] dx = \int_{\Gamma(\theta)} V.\nu d\gamma \tag{51}$$

and

$$\begin{aligned} L_4''(\theta)(\xi,\eta) &= \int_{\omega} (tr[DV]tr[DW] - tr[DV.DW]) dx \\ &= \int_{\Gamma(\theta)} V_i (\delta_i W_i \nu_i - \delta_i W_j \nu_j) d\gamma \end{aligned} \tag{52}$$

Finally, adding (35),(45),(49) and (51) we obtain (32) and adding (44),(46),(50) and (52) we obtain (33).

According to (32) the variational formulation of the continuous problem (2)-(7) consists in finding (θ^*, Λ^*) such that:

$$L'(\theta^*, \Lambda^*)(\xi) = 0 \tag{53}$$

for every $\xi \in \mathcal{O}^{-1}$ admissible.

The fact that $L'(\theta, \Lambda)$ and $L''(\theta, \Lambda)$ are a boundary integral equation makes numerical approach possible.

4 Numerical Method.

4.1 Discretization.

We want to evaluate numerically an approximation of the optimal domain Ω^* such that the discretized shape first order necessary condition (32) vanishes for all admissible vector field. To this end we construct a sequence of domains $\omega(\theta^k)$, more precisely, we consider a sequence of domains defined by their boundaries $\Gamma^k = \partial\omega(\theta^k)$ that converge towards a critical point. By Γ^k we mean a piecewise linear closed surface, that is, a union of triangles T_i in $I\!R^3$. The nodes of the surface Γ^k are denoted by $x^{i,k}$.

At each vertex $x^{i,k}$ of Γ^k is associated a direction $\hat{Z}^{i,k} \in I\!R^3$. We construct a continuous piecewise linear vector field $Z^{i,k}$ from Γ^k in $I\!R^3$ such that $Z^{i,k}(x^{j,k}) = \delta_{i,j}\hat{Z}^{i,k}$.

The support of $Z^{i,k}$ is equal to the union of the triangles for which $x^{i,k}$ is a node. At each iteration we compute the following vector field:

$$Z^k(x) = \sum_{i=1}^{n} u_i Z^{i,k}(x) \tag{54}$$

and the updated surface Γ^{k+1} is then given by:

$$\Gamma^{k+1} = \left\{X = x + \sum_{i=1}^{n} u_i Z^{i,k}(x); u_i \in I\!R, x \in \Gamma^k\right\} \tag{55}$$

where $\bar{u}^t = (u_1, \cdots .u_n) \in I\!R^n$ are the unknowns which determine the evolution of the surface Γ^k.

This method of evolution of the boundary has the important advantage that there is only one degree of freedom at each node.

4.2 Integral equation.

To compute the shape gradient we solve at each iteration the exterior Neumann problem (8). Due to theorem 2.1 we solve numerically the second kind Fredholm equation (7) by a Galerkin method.

If we note $\overline{q}(x)$ an approximation of $q(x)$ such that $q \in C^0(\Gamma^k)$ and is piecewise linear, then

$$\overline{q}(x) = \sum_{i=1}^{n} q_i \Psi_i(x) \tag{56}$$

where $\Psi_i(x_j) = \delta_{i,j}$ and is linear in each triangle.

Then we obtain $\bar{q}^t = (q_1, \cdots, q_n) \in I\!R^n$ as the solution of the linear system:

$$A\bar{q} = \bar{b} \tag{57}$$

where:

$$\begin{aligned} a_{i,j} &= \int_{\Gamma^k \times \Gamma^k} \Psi_i(x)\Psi_j(y) \frac{(\nu(x), x-y)}{|x-y|^3} d\gamma(x) d\gamma(y) \\ &+ 2\pi \int_{\Gamma^k} \Psi_i(x)\Psi_j(x) d\gamma(x) \end{aligned} \tag{58}$$

and

$$b_i = -4\pi \int_{\Gamma^k} (B^1(x) \cdot \nu(x)) \Psi_i(x) d\gamma(x) \tag{59}$$

The matrix A is a general, non-symmetric and dense thus the LU factorization is used to solve (57). The $a_{i,j}$ coefficients are numerically computed by a Newton-Cotes method using $k \times k$ quadrature points. The support of the basis functions $\Psi_i(x)$ is limited to the triangles which have $x^{i,k}$ as a vertex. In our discretization the number of triangles associated to a vertex $x^{i,k}$ is six in almost all the cases. Then the evaluation of the $a_{i,j}$ coefficients by a quadrature formula needs $6 \times k \times k \times n^2$ floating point computations of the integrand.

The LU factorization of A needs $\frac{2n^3}{3}$ floating points operations because A is dense while the subsequent forward-backward solution requires $2n^2$ flops.

The implementation of the Newton method requires at each iteration the numerical solution of the exterior Neumann system (31) when we replace V by $Z^k(x) = \sum_{i=1}^n u_i Z^{i,k}(x)$.

Then we set:

$$\varphi'_{Z^k}(x) = \sum_{i=1}^{n} u_i \varphi'_{Z^{i,k}}(x) \tag{60}$$

and we obtain n systems similar to (8). The only difference are the second members of the equations because the boundary Neumann condition is not the same for each $Z^{i,k}(x)$. Thus the numerical solution of the n systems requires $2n^3$ flops because the LU factorization of A is known at this step of the algorithm.

4.3 The Numerical Algorithm.

To analyse the algorithm we give a scheme of the computational process.

1. We give an initial shape Γ^0 and a Lagrangian multiplier Λ^0.
2. For $k = 0, ...$ to convergence do :
   a) Compute the perturbation direction basis $Z^{i,k}$, $i = 1, ..., n$.
   b) Solve the integral equation (10).
      b.1 - Compute the coefficients of the matrix A.
      b.2 - Compute b_i, $i = 1, \cdots, n$.
      b.3 - Factorize the non symmetric matrix A.
      b.4 - Solve the linear system to compute $\overline{q}$.
   c) Compute a numerical approximation of the first shape derivative.
      c.1 - Compute $\nabla\varphi$ and B^1 at all the quadrature points of the surface Γ^k.
      c.2 - Calculate numerically the integral (32) when $V = Z^{i,k}, i = 1, \cdots, n$.
      c.3 - Compute the volume $m(\omega^k)$ and its shape derivative.

c.4 - Calculate a numerical approximation of
$D(\Gamma^k, \Lambda^k)(Z^{i,k}), i = 1, \cdots, n.$

d) Compute a numerical approximation of the second order shape derivative.
$L''(\Gamma^k)(Z^{i,k}, Z^{j,k})$ for $i, j = 1, \cdots, n.$

d.1 - Compute $\nabla \varphi'_{Z^{i,k}}, i = 1, \cdots, n$ at all the quadrature points.

d.2 - Calculate the integral (33).

e) Solve the symmetric Newton system(26) to compute
$\overline{u}^t = (u_1, \cdots, u_n) \in I\!R^n.$

f) Update the surface Γ^k to obtain Γ^{k+1}.

We will study the complexity of the algorithm as function of the number of nodes needed for the piecewise linear representation of the surface Γ^k (n is also the number of degrees of freedom of the problem).

First we look for the number of flops required to compute de gradient. In section(4.2) we remark that the evaluation of $a_{i,k}$ by a quadrature formula needs $6 \times k \times k \times n^2$ flops and the numerical solution of the linear system (57) requires $\frac{2n^3}{3} + 2n^2$ flops. Thus, setting $k = 3$ the numerical solution of the integral equation (10) needs $O(n^3)$ flops.

The support in Γ^k of the perturbation vector field $Z^{i,k}(x)$ is exactly the same than the basis element $\psi_i(x)$. Then the numerical computation of the gradient coefficient

$$\frac{1}{2\mu_0} \int_\Gamma (B^2 + \sigma \mathcal{H} + \rho g x_3 - \Lambda)(Z^{i,k} \cdot \nu) d\gamma \tag{61}$$

requires $O(n)$ flops. Consequently the gradient evaluation adds $O(n^2)$ flops. Summarizing, the number of flops needed to perform step a), b) and c) of the algorithm are of order $O(n^3)$.

The step d) is devoted to the evaluation of the Hessian. The computation of the Hessian needs $\frac{n(n+1)}{2}$ computations of the second order shape derivative $L''(\Gamma^k)(Z^{i,k}, Z^{j,k})$, $i = 1, ..., n, j = i, ..., n$.

The Hessian matrix coefficients are given in (33) when V is replaced by $Z^{i,k}$ and W by $Z^{j,k}$.

The boundary integral $L''(\Gamma^k)(Z^{i,k}, Z^{j,k})$ is composed of two terms. The first one is classical in finite element calculations, needs $O(n)$ flops.

The second term which is not zero for all i and j is the following:

$$\begin{array}{c} \int_{\Omega(\theta^k)} \nabla\varphi'(\theta)(\xi)\cdot\nabla\varphi'(\theta)(\eta)dx \\ = \\ \int_{\Omega^k} \nabla\varphi'_{Z^{i,k}}\cdot\nabla\varphi'_{Z^{j,k}}dx. \end{array} \tag{62}$$

By the Green formula and using the fact that φ' is solution of (31) we obtain the following boundary integral form:

$$\begin{array}{c} \int_{\Omega^k} \nabla\varphi'_{Z^{i,k}}\cdot\nabla\varphi'_{Z^{j,k}}dx = -\int_{\Gamma^k}\varphi'_{Z^{i,k}}\cdot\frac{\partial\varphi'_{Zj,k}}{\partial\nu}d\gamma(x) \\ = -\int_{\Gamma^k}(Z^{i,k}\cdot\nu)(\nabla\varphi + B_1)\cdot(\delta\varphi'_{Z^{j,k}})d\gamma(x). \end{array} \tag{63}$$

In order to compute (33), it will be necessary to evaluate $\nabla\varphi'_{Z^{i,k}}$, thus we have to solve the n exterior Neumann problems (31). In section (4.2) we remark that the solution of the n systems requires $2n^3$ flops.

Having solved the n exterior Neumann problems the evaluation of (33) adds $O(n)$ flops.

In conclusion we have that the numerical computation of the Hessian needs only $O(n^3)$ floating points operations.

The resolution of the linear system (26) by a LDL^t numerical matrix decomposition needs also only $O(n^3)$ flops, see [1]. Thus the complexity of a Newton iteration is similar to the complexity of a gradient method, namely $O(n^3)$.

In fact, at each iteration, solving the integral equation takes about forty five percent of the C.P.U. time in a sequential computer. The table 4.1 gives the percent of the most C.P.U. time consuming sections of the algorithm in a sequential mode using an example where $n = 1024$.

TABLE 1: Example, n=1024.

	$a_{i,j}$	LU	$\nabla\varphi$	$\nabla\varphi'$	L''
%	37	3.4	7.6	32	13

Consequently to take advantage of the relative independence between the coefficients of the gradient and the coefficients of the Hessian matrix, we introduce a parallel version of the code using message-passing on a MIMD computer.

5 Parallel implementation.

The main problem in a parallel approach using the message-passing programming model in MIMD computers is the impact on the total time of the C.P.U. time spent in interprocessor communications. Then the goal is to obtain an algorithm with an optimal ratio between communication cost and floating point operations cost.

To determine the theoretical performances of the parallel code we introduce a simple model of the speedup behaviour.

We denote by t_t the amount of time needed to compute a floating point operation, by t_c the time needed to communicate a floating point number.

The sequential C.P.U. total time is denoted by t_s and we assume that $t_s = O_t(1)n^t t_t$. Let $t(N_p)$ be the total time when we run the code in N_p process MIMD computers. We assume $t(N_p) = \frac{t_s}{N_p} + O_c(1)n^c t_c$.

Then the speedup $s(N_p) = \frac{t_s}{t(N_p)}$ can be modeled as:

$$s(N_p) = \frac{O_t(1)n^t t_t}{\frac{O_t(1)n^t t_t}{N_p} + O_c(1)n^c t_c} = N_p \frac{1}{1 + N_p \frac{O_c(1)}{O_t(1)} \frac{t_c}{t_t} n^{c-t}} \tag{64}$$

We assume that the ratio $\frac{t_c}{t_t}$ is independent of n.

Then if $c - t < 0$ the speedup increases with the number of unknowasn and has an asymptotic maximum N_p. If $c - t > 0$ the speedup decreases when $n \to \infty$. If $c - t = 0$ the speedup depends on the ratio $\frac{O_c(1)}{O_t(1)}$.

We denote by $e(N_p)$; $e(N_p) = \frac{s(N_p)}{N_p}$ the efficiency. Then to obtain a maximum of efficiency when we parallelize the section of the algorithm where $c - t \leq 0$.

5.1 Computation of the matrix A.

The algorithm of sequential computation of the $a_{i,j}$ coefficients for $i, j = 1, 2, ..., n$ is a classical finite element routine to "assemble" the stiffness matrix

A. We denote by $n\_element$ the number of finite elements used to represent Γ^k. In the sequential case the code takes the following form:

do $ey = 1, n\_element$
  do $ex = 1, n\_element$
C if $x^{i,k} \in T_{ex}$ and $x^{j,k} \in T_{ey}$ then
  $A(i,j) = A(i,j) + \int_{T_{ex}} \int_{T_{ey}} \Psi_i(x)\Psi_j(y)\frac{(\nu(x),x-y)}{|x-y|^3} d\gamma(x)d\gamma(y)$
C if $ex = ey$
  $A(i,j) = A(i,j) + \int_{T_{ex}} \int_{T_{ey}} \Psi_i(x)\Psi_j(y)\frac{(\nu(x),x-y)}{|x-y|^3} d\gamma(x)d\gamma(y)$
  $+2\pi \int_{\Gamma^k} \Psi_i(x)\Psi_j(x)d\gamma(x)$
  enddo
enddo

Following the message_passing programming model, we introduce for A a scattered column data structure. The columns of A are distributed across the processors following a one dimensional static wrap mapping scheme. Let n denote the number of given equations. Each column j of A denoted A_j is stored in the processor number $p = mod(j-1, N_p)$.

Then we propose the following parallel version using M.P.I.:

C each processor runs the following code
do $ey = myid, n\_element, N_p$
  do $ex = 1, n\_element$
C if $x^{i,k} \in T_{ex}$ and $x^{j,k} \in T_{ey}$,
  $A(i,j) = A(i,j) + \int_{T_{ex}} \int_{T_{ey}} \Psi_i(x)\Psi_j(y)\frac{(\nu(x),x-y)}{|x-y|^3} d\gamma(x)d\gamma(y)$
C if $ex = ey$
  $A(i,j) = A(i,j) + \int_{T_{ex}} \int_{T_{ey}} \Psi_i(x)\Psi_j(y)\frac{(\nu(x),x-y)}{|x-y|^3} d\gamma(x)d\gamma(y)$
  $+2\pi \int_{\Gamma^k} \Psi_i(x)\Psi_j(x)d\gamma(x)$
  enddo
enddo

At this stage each processor has an incomplete copy of A. In the next part of the algorithm we complete at each processor the computation of the coefficient of his own columns.

do $j = 1, n$
  its_j= the number of the processor owner of j-th column
  if $myid = its\ _j$ then
  receive the column A_j and update
  the values of coefficients $a_{i,j}$ for all i.
  $A_j = A_j + A_j$
  else
  Send the column A_j to process its_j
  endif
enddo

This algorithm needs $O(n^2)$ communications but does not repeat any calculations. The theoretical speedup is given by:

$$s(N_p) = \frac{O_t(1)n^2 t_t}{\frac{O_t(1)n^2 t_t}{N_p} + O_c(1)n^2 t_c} = N_p \frac{1}{1 + N_p \frac{t_c}{t_t} \frac{O_c(1)}{O_t(1)}} \tag{65}$$

In our case $O_t(1)$ is the number of floating point operations needed to compute the integrand of the $a_{i,j}$ coefficient form at a quadrature point.

5.2 LU decomposition.

We factorize the matrix A by a standard Gauss decomposition with a partial pivot searching. Thus the process myid will execute the code:

do $k = 1, n$
  if $mod(k, myid + 1) = 0$ then
  .find the partial pivot
  .make the permutation if necessary
  .broadcast the column A_k and the index pivot
  endif
  myj=min{ j such that $j.N_p + myid + 1 \geq k + 1$ }
  do $j = myj, n, N_p$
  perform the elimination step in column j
  enddo

enddo

The amount of time needed to factorize A in N_p processors is given by $t(N_p) = \frac{t_s}{N_p} + n^2 t_c$ then the speedup :

$$s(N_p) = N_p \frac{1}{1 + 3N_p \frac{n^2}{n^3} \frac{t_c}{t_t}} \tag{66}$$

converges to N_p when $n \to \infty$.

At this step, each processor knows its own column of U and all the matrix L. Using this distribution of the data we perform the solution of the two linear systems $LUq = b$ by using a classical parallel algorithm, see [16], [18].

5.3 The Hessian computation.

Before computing the Hessian we solve the n exterior problems(31). In section (4.2) we remarked that the matrix of the linear system associated to the discretization of (31) is the same for all problems. Only the right hand side differs between two exterior problems. Then we make use of the same LU decomposition to solve in parallel the n problems. To this end we broadcast the U matrix to all the processors and we solve with each processor $2n/N_p$ linear triangular systems.

Once all the linear systems are solved in parallel, we broadcast the results to all the processes. In conclusion we need $O(n^2)$ transmissions and $O(n^3)$ flops to compute and distribute the results of the n exterior systems (31).

The data structure adopted for the Hessian H is the same as the one used for A. Each column H_j is stored in the processor number $p = mod(j-1, N_p)$. In fact we separate the numerical structure of H two different parts, the first one non sparse and second the one sparse, this last one depending on the perturbation vector field $Z^{i,k}$. Then first we compute the non sparse part of H, each processor running then the following code:

do $j = myid, n, N\_p$
  do $i = 1, n$
  $H(i,j) = -\int_{\Gamma^k} (Z^{i,k} \cdot \nu)(\nabla\varphi + B^1)(\delta\varphi'_{Z^{i,k}} d\gamma(x)$

```
      enddo
   enddo
```

The other terms of the integral (33) are computed using the scheme used to "assemble" A. Each processor running now the following algorithm:

```
      do 100 ey = 1, n_element
C let be J = {j/x^{j,k} ∈ T_ey}
      If J ⋂{ the set of column of H stored in myid processor }
            go to 100
      else
            do ex = 1, n_element
C let be I = {i such that x^{i,k} ∈ T_ex}
            for i ∈ I and j ∈ J do
            H(i,j) = H(i,j)+all the others term in the integral (33).
            enddo
            endif
      enddo
```

This procedure needs $O(n^2)$ flops. Then the total computation of H needs $O(n^2)$ communications and $O(n^3)$ flops. Then the speedup goes to N_p when $n \to \infty$.

Because the data structure of H is the same as that of A we perform a parallel LDL^t matrix decomposition of H using the same methodology than the one used in the LU decomposition of A.

6 Numerical results.

This section will be devoted to the analyse of various examples in shape optimization.

We write the energy (7) as:

$$E(\omega) = B_m \int_{\Omega(\theta)} B^2(\theta) dx + \sigma \int_{\Gamma(\theta)} d\gamma + B_g \int_{\omega(\theta)} x_3 dx$$

where the constant B_m, σ and B_g are given constant. In all the cases studied the current density j_0 is non zero only in one dimensional wires.

In all the cases we observe the evolution of $\|\Gamma^* - \Gamma^k\|_{L^2}$ which is the L^2 error between a 'right solution' Γ^* obtained after a great number of iterations and Γ^k, the domain at iteration k.

We also study the evolution of the norm of the Lagrangian shape gradient $\|L'\|_{L^2}$, $|m_k - m_0|$ the error between the volume of the domain at iteration k and the given volume m_0.

We start with a discretize sphere which is considered as a first rough approximation of the free surface, see figure 6.1.

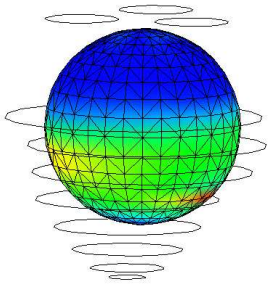

FIGURE 6.1: Initial free surface for the first example.

The first example concerns a bubble where the magnetic field B is created by eleven wires in a non symmetric configuration.

We present two numerical cases. The first one with 466 nodes converges after 10 iterations. In the second case we discretize the free surface with 722 nodes and 1440 triangles. The free surface obtain is non symmetric. In figure 2 we plot the evolution of the $\|L'\|_{L^2}$ in the two cases, with a logarithmic scale.

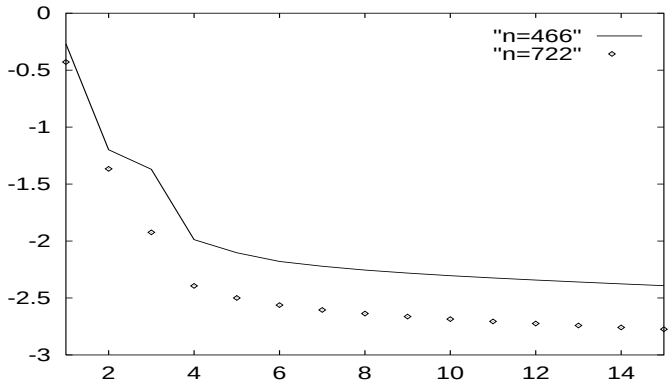

FIGURE 6.2: Evolution of the $\|L'\|_{L^2}$ in the cases $n = 466$ and $n = 722$ during 15 iterations.

l	$\|\Gamma^* - \Gamma^k\|_{L^2}$	$\|L'\|_{L^2}$	$\|m_0 - m_k\|$	l	$\|\Gamma^* - \Gamma^k\|_{L^2}$	$\|L'\|_{L^2}$	$\|m_0 - m_k\|$
1	0.28160745	0.538742	0.000015	1	0.25110406	0.372630	0.000002
2	0.18938252	0.063276	0.067681	2	0.16177495	0.042990	0.068132
3	0.06923706	0.042613	0.095329	3	0.07439711	0.011917	0.056972
4	0.04567913	0.010282	0.002560	4	0.04414708	0.004039	0.002251
5	0.03178702	0.007892	0.000690	5	0.02783905	0.003162	0.000682
6	0.02448614	0.006620	0.000131	6	0.02026516	0.002739	0.000240
7	0.02172659	0.006003	0.000040	7	0.01508798	0.002484	0.000101
8	0.01944490	0.005557	0.000011	8	0.01145004	0.002307	0.000040
9	0.01717505	0.005230	0.000003	9	0.00881417	0.002172	0.000021
10	0.01495014	0.004965	0.000007	10	0.00684190	0.002062	0.000010
11	0.01274491	0.004741	0.000011	11	0.00531275	0.001969	0.000013
12	0.01055950	0.004545	0.000010	12	0.00408282	0.001887	0.000005
13	0.00839523	0.004368	0.000016	13	0.00305502	0.001813	0.000000
14	0.00625499	0.004207	0.000005	14	0.00216839	0.001745	0.000000
15	0.00414133	0.004058	0.000017	15	0.00138062	0.001683	0.000007

TABLE 6.1: First example. The $\|\Gamma^* - \Gamma^k\|_{L^2}$, $\|L'\|_{L^2}$, $|m_0 - m_k|$ values for 466 and 722 nodes.

In figures 6.3 – 6.6 we plot the evolution of the free boundary for the first example. The first iteration are critical to achieve a good approximation of the free surface. This shape is the result of the contradictory action of the magnetic pressure, the surface tension forces and the gravity. The result is a non symetric shape.

The figure 6.2 shows that the rate of convergence is superlinear at the first iterations and it becames linear after four Newton iteration. Also we observe

that we obtain a better approximation of the Euler necessary condition with 722 nodes.

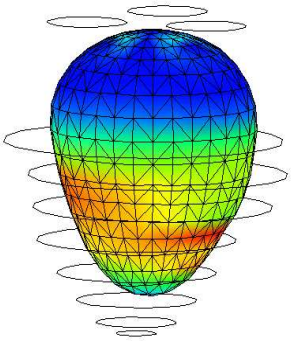

FIGURE 6.3: Itération 01.

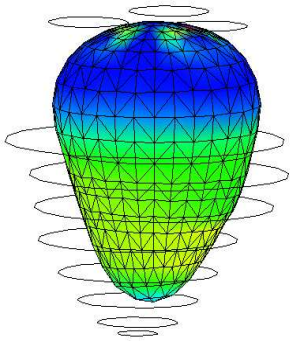

FIGURE 6.4: Itération 02.

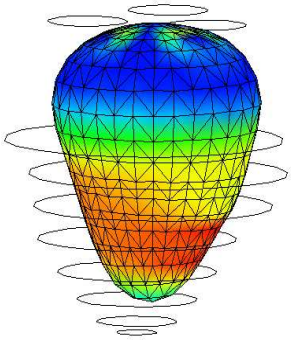

FIGURE 6.5: Itération 03.

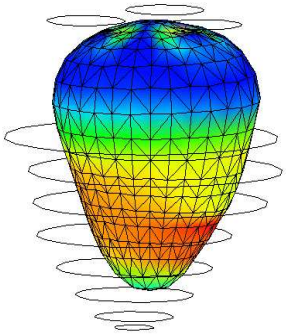

FIGURE 6.6: Itération 15.

The second example, (see figure 6.7), concerns a bubble where the magnetic field is created by 10 wires with a non symmetric configuration.

In table 6.2 we observe the behaviour of $\|\Gamma^* - \Gamma^k\|_{L^2}$ which is the L^2 error between a "right solution" Γ^* obtain after a great number of iterations and a very fine discretisation of the free boundary. The $\|L'\|_{L^2}$ rate of convergence to zero is superlinear the first iteration and it becames linear after four iterations. In figure 6.8 we follows the behaviour of $\|L'\|_{L^2}$ during 15 iterations.

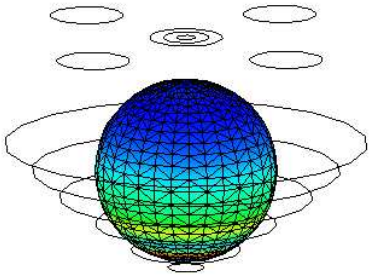

FIGURE 6.7: Initial free surface for the second example.

l	$\|\Gamma^* - \Gamma^k\|_{L^2}$	$\|L'\|_{L^2}$	$\|m_0 - m_k\|$
1	0.550346136093	0.18274675	0.00000858
2	0.363416433334	0.03846094	0.14281559
3	0.234483063221	0.01779965	0.14949131
4	0.166946530342	0.00816778	0.02791786
5	0.099322676659	0.00461759	0.01086760
6	0.047078490257	0.00309315	0.00307322
7	0.020525963977	0.00248215	0.00065947
8	0.014611248858	0.00217972	0.00006342
9	0.011999816634	0.00199168	0.00003862
10	0.009782091714	0.00185390	0.00002670
11	0.008058317937	0.00174668	0.00001287
12	0.006706404500	0.00166098	0.00002193
13	0.005340330768	0.00159163	0.00000334
14	0.003977795132	0.00153530	0.00000191
15	0.002631784417	0.00148988	0.00001240

TABLE 6.2: Second example. The evolution of the $\|\Gamma^* - \Gamma^k\|_{L^2}$, $\|L'\|_{L^2}$ and $|m_0 - m_k|$ in the case of 722 nodes and 1440 finite element.

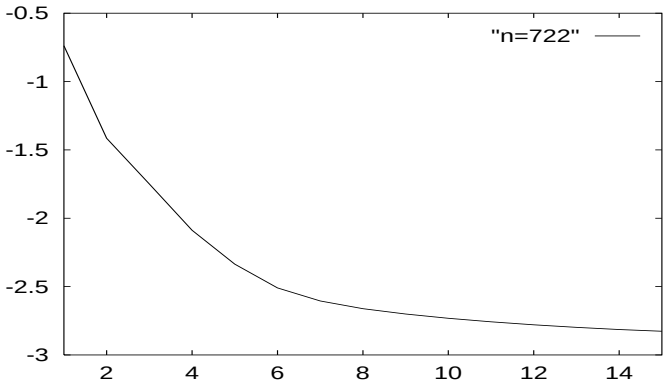

FIGURE 6.8: Behaviour of the $\|L'\|_{L^2}$ in the case of 722 nodes and 1440 finite element.

In figures 6.08 – 6.11 we plot the free boundaries achieve at intermediate iterations. In the first iterations we observe a finite deformation, the following iterations are devoted to fit the free surface.

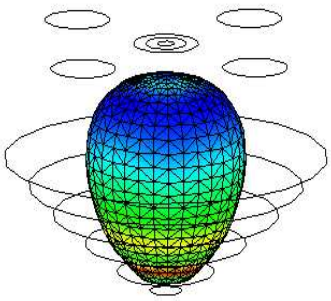

FIGURE 6.8: Itération 01.

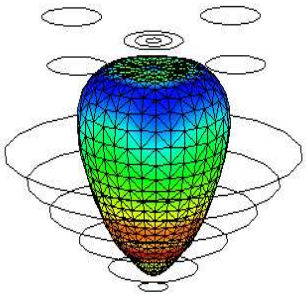

FIGURE 6.9: Itération 02.

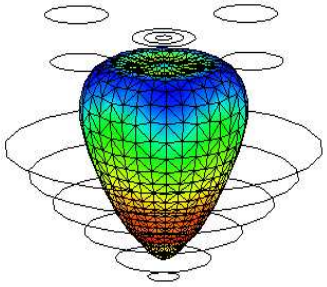

FIGURE 6.10: Itération 03.

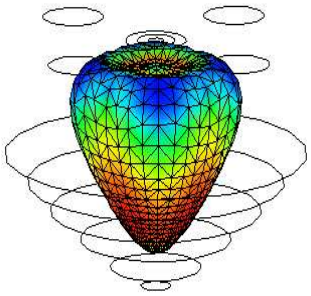

FIGURE 6.11: Itération 15.

Both examples studied in this section shows that we have a superlinear rate of convergence at first iterations. After that the effects of errors is that the residual $\|L'\|_{L^2}$ is no longer reduced in a significative manner as stagnation of the iterations. In fact instead of $L'(\theta^k)$ and the Hessian $H(\theta^k, \Lambda^k)$ we compute a numerical approximation $L'(\theta^k, \Lambda^k) + \delta^k$ and $H(\theta^k, \Lambda^k) + \eta^k$. If $||\delta^k||_{L^2} = \epsilon$ is due to floating-point roundoff, there is no reason to expect that $||L'||_{L^2}$ will be smaller than ϵ in general, see [23]. Thus the numerical computation of the solution of a 3 dimensional shape optimisation problem at low cost is possible using a Newton method because the complexity of a iteration is similar to the complexity of a Quasi-Newton method, namely $O(n^3)$.

6.1 Efficiency Results.

In this section we report the efficiency results obtain in the SILICON GRAPHICS PowerChallenge Array R10000 at 180Mhz of the High Performace Computer Center Charles Hermite. Table 6.3 the efficiency of code computing the coefficients $a_{i,j}$ are reported when we consider 1,2,4 and 7 processors with different numbers of degrees of freedom. We observe that the results are better than the theoretical prevision, we obtain an efficiency near 1.

n	1	2	4	7
466	1.00	0.95	0.92	0.90
594	1.00	0.95	0.93	0.90
722	1.00	0.96	0.95	0.925
850	1.00	0.98	0.95	0.935
1042	1.00	0.99	0.98	0.95

TABLE: 6.3: Example 2, Efficiency of parallel code in computation of A.

In table 6.4 we report the results obtained in the parallelisation of the LU matrix decomposition. Here the maximum efficiency obtained is 0.75 because of communication cost.

n	1	2	4	7
466	1.00	0.87	0.67	0.51
594	1.00	0.91	0.77	0.60
722	1.00	0.91	0.80	0.61
850	1.00	0.94	0.82	0.65
1042	1.00	0.96	0.87	0.75

TABLE 6.4: Example 2, Efficiency of parallel LU decomposition code.

In table 6.5 are reported the results of the code computing the second order shape derivative of the Lagrangian. We obtain an efficiency near 1 which is consistent with the theoretical previsions.

n	1	2	4	7
466	1.00	0.98	0.98	0.96
594	1.00	0.98	0.98	0.96
722	1.00	0.99	0.99	0.97
850	1.00	0.99	0.99	0.98
1042	1.00	0.99	0.99	0.988

TABLE 6.5: Example 2, Efficiency of parallel code in computation L".

In fact each step of the code was parallelized using the message-passing programming model, using MPI interface in the SILICON GRAPHICS Power-Challenge Array R10000 at 180Mhz. Table 6.6 reports the efficiency obtained when we run all the code.

n	1	2	4	7
466	1.00	0.95	0.86	0.77
594	1.00	0.97	0.89	0.77
722	1.00	0.99	0.89	0.80
850	1.00	0.99	0.89	0.80
1042	1.00	0.99	0.91	0.81

TABLE 6.6: Example 2, Global Efficiency of the parallel code.

In summary the efficiency results presented here show that considering Newton's like algorithm in shape optimization allow us to take a maximal advantage of the architecture of the MIMD computers. There is one reason to explain this fact: the complexity in the floating point approximation of the Hessian and the gradient is dominated by the complexity of the algorithm solving the P.D.E.'s. and we can parallelize very efficiently the P.D.E.'s solving code and the Hessian evaluation code.

Acknowledgement. This work has been supported by the High Performace Computer Center Charles Hermite of Nancy.

References

[1] G. H. Golub and C. F. Van Loan, *Matrix Computations. 2nd ed.*, The Johns Hopkins Press, Baltimore, MD., 1989.

[2] R. Kress, *Linear Integral Equations*, Springer Verlag, Berlin, Heilderberg, 1989.

[3] R. Fletcher, *Practical Methods of Optimization*, John Wiley & Sons, Chichester, 1987.

[4] J. Sokolowski and J.P. Zolesio, *Introduction to Shape Optimization,Shape Sensitivity Analysis*, Springer Verlag, Berlin, Heilderberg, 1992.

[5] J.Sokolowski, *Displacement Derivatives in Shape Optimization of Thin Shells*, Contemporary Mathematics, Vol 209, pp. 247-266, 1997.

[6] J.R. Roche, *Algorithmes numériques en optimisation de formes et électromagnétisme*, Mémoire d'Habilitation à Diriger des Recherches, Université Henri Poincaré, Nancy 1, 1996.

[7] J.R. Roche, *Gradient of the discretized energy method and discretized continuous gradient in electromagnetic shaping simulati on*, Appl. Math. and Comp. Sci., Vol. 7, $n°3$, pp. 545-565. 1997.

[8] A. Novruzi *Contribution en Optimisation de Formes et Applications.* Thèse de l'Université Henri Poincaré, Nancy 1, 1997.

[9] A. Novruzi and J.R. Roche, *Second Order Derivatives, Newton Method, Application to Shape Optimization*, Rapport INRIA, 1995.

[10] J.-C. Nedelec, *Approximation des équations intégrales en mécanique et en physique*, Rapport Interne, Ecole Polytechnique, 1977.

[11] S. Murat and J. Simon, *Sur le contrôle par un domaine géométrique*, Rapport du Laboratoire d'Analyse Numérique, Université de Paris , 1976.

[12] J. Simon, *Differentation with respect to the domain in boundary value problems*, Numer.Funct.Anal.And.Opt, 2, (1980) pp 769.

[13] Y. Goto and N. Fujii, *Second order numerical method for domain optimization problems*, Journal of Optimization Theory and Applications, 67,(1990) pp. 533-550.

[14] A. Henrot and M. Pierre, *Un probléme inverse en formage de métaux liquides*, M^2AN, 23,(1989) pp. 155-177.

[15] M. Pierre and J.R. Roche, *Numerical simulation of tridimensional electromagnetic shaping of liquid metals*, Numer. Math, 65,(1993) pp. 203-217.

[16] C. Farhat and F.-X. Roux, *Implicit parallel processing in structural mechanics*, Computational Mechanics Advances, 2, $n°1$ (1994) pp.1-124.

[17] W. Gropp, E. Lusk and A. Skjellum, *Using MPI, Portable Parallel Programming with the Message-Passing Interface.* The MIT Press, Cambridge, Massachusetts, 1994.

[18] E.F. Van de Velde, *Concurrent Scientific Computing*, Springer-Verlag, New York, 1994.

[19] R. Dautray and J.-L. Lions, *Analyse mathématique et calcul numérique*, Masson, Paris, 1988.

[20] E. Giusti, *Minimal Surfaces and Functions of Bounded Variation*, Birhäuser, Boston, 1984.

[21] O. Coulaud and A. Henrot, *A nonlinear boundary value problem solved by spectral methods*, Appl. Anal., 43, (1991) pp. 229-244.

[22] S.G. Nash and A. Sofer, *Block Truncated-Newton Methods for Parallel Optimization*, Mathematical Programming, Ser. B 45, $n°1$ (1989) pp. 529-546.

[23] T.J. Ypma, *The effect of rounding errors on Newton-like methods*, IMA J. Numer. Anal., 3, (1983) pp. 109-118.

[24] A.S. Zenios and J.M. Mulvey, *Vectorization and multitasking of nonlinear network programming algorithms*, Mathematical Programming, Ser. B 42, $n°2$ (1988) pp. 449-470.

[25] A.S. Zenios and M.C. Pinar, *Parallel block-partitioning of truncated Newton for nonlinear network optimization*, SIAM Journal on Scientific and Statistical Computing, 12, $n°5$ (1992) pp. 1173-1193.

Unit´e de recherche INRIA Lorraine, Technopˆole de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhˆone-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399